A note on the calculation of step-lengths in interior-point methods for semidefinite programming (Continuous and Discrete Mathematics for Optimization)

AUTHOR(S):

Toh, Kim-Chuan

CITATION:

Toh, Kim-Chuan. A note on the calculation of step-lengths in interior-point methods for semidefinite programming (Continuous and Discrete Mathematics for Optimization). 数理解析研究所講究録 1999, 1114: 106-113

ISSUE DATE:

1999-11

URL:

http://hdl.handle.net/2433/63392

数理解析研究所講究録
1114巻 1999年 106-113

A note on the calculation of step-lengths in interior-point methods for semidefinite programming

Kim-Chuan Toh *

Abstract

In each iteration of an interior-point method for semidefinite programming, the maximum step-length that can be taken by the iterate while maintaining the positive semidefiniteness constraint need to be estimated. In this note, we show how the maximum step-length can be estimated via the Lanczos iteration, a standard iterative method for estimating the extremal eigenvalues of a matrix. We also give a posteriori error bounds for the estimate.

1 Introduction

Suppose X is an $n \times n$ symmetric positive definite matrix that is the current primal iterate of an interior point method in semidefinite programming (SDP), and ΔX is the search direction to be taken. To define the next iterate, we need to estimate the maximum value the step-length α can take while satisfying the positive semidefiniteness condition $X + \alpha \Delta X \succeq 0$. In a primal-dual interior-point method, the same need to be done for the dual variable Z, but we shall concentrate only on the primal variable X in this note. We refer the reader to [1], [3], [8], [10], [16], among others, for details on interior-point methods for SDP.

Let $\alpha_{\max}$ denotes the maximum allowed step-length. So far as we are aware of, there are two methods used to estimate $\alpha_{\max}$ in all the current implementations of

*Department of Mathematics, National University of Singapore, 10 Kent Ridge Crescent, Singapore 119260. (mattohkc@math.nus.edu.sg). Research supported in part by National University of Singapore Academic Research Grant RP972685.

interior-point methods for SDP; see [2], [5], [6], [7], [8], [14] and [17]. The first method to estimate $\alpha_{\max}$ is to check the positive semidefiniteness condition via Cholesky factorization by backtracking. That is, starting with $\alpha = 1$, the method successively reduce α by a fixed factor (say, 0.8) until the condition is satisfied. The resulting value of α is then a lower bound for $\alpha_{\max}$.

The second method is to compute $\alpha_{\max}$ exactly by finding the maximum eigenvalue of an $n \times n$ symmetric matrix via the QR algorithm [9]. The mechanism is as follows. Suppose $X = R^T R$ is the Cholesky factorization of X. Let

$$B = -R^{-T} \Delta X R^{-1}.$$

Then it is readily shown that the condition $X + \alpha \Delta X \succeq 0$ is equivalent to $I - \alpha B \succeq 0$. In other words, α times the maximum eigenvalue of B should be no greater than 1. Thus when $\lambda_1(B)$, the maximum eigenvalue of B, is positive, we have

$$\alpha_{\max} = \frac{1}{\lambda_1(B)}. \tag{1}$$

(Note that if $\lambda_1(B)$ is non-positive, then $\alpha_{\max}$ can be chosen to be any positive real number.)

The first method requires $Kn^3/3$ flops, where K is the number of Cholesky factorization performed in the backtracking process. The second method requires a total of $3n^3$ flops in forming the symmetric matrix B and computing $\lambda_1(B)$ via the QR algorithm. Typically, the first method is much cheaper since K is usually no greater than three when the reduction factor in the backtracking process is fixed at a reasonable value such as 0.8. The main drawback is that its estimate of $\alpha_{\max}$ is too conservative, which sometimes unnecessarily dampen the progress of the interior-point iteration. On the other hand, the second method gives accurate $\alpha_{\max}$, but it is too costly. From our computational experience on the SDPLIB problems [4] using the MATLAB software SDPT3 version 1.3 [17], calculating the step-lengths using the second method for both the primal and dual variables in a Mehrotra predictor-corrector interior-point algorithm can sometimes take up to 40% of the total CPU time spent in each interior-point iteration.

Now observe that in order to estimate $\alpha_{\max}$ accurately, all we need is to have a tight upper bound ρ for $\lambda_1(B)$. It is not necessary to know the exact value of $\lambda_1(B)$ in order to ensure that $X + \alpha \Delta X \succeq 0$. (Note that if ρ is non-positive, then again $\alpha_{\max}$ can be chosen to be any positive real number.) This flexibility gives us the freedom to use estimates of $\lambda_1(B)$ in estimating $\alpha_{\max}$. We propose to estimate $\lambda_1(B)$ by using the Lanczos iteration, which is a well known iterative method in numerical

linear algebra for finding the extremal eigenvalues of a matrix. But since iterative methods for estimating eigenvalues of matrices are not generally well known to the optimization community, the purpose of this note is to bring them to the attention of the community. We show that the Lanczos iteration can be applied nicely to estimate the maximum step-length allowed in each iteration of an interior-point for SDP, and provide computable error bounds on the accuracy of the estimation.

The main advantage of using the Lanczos iteration to estimate $\lambda_1(B)$ lies in the fact that only $\mathcal{O}(n^2)$ flops are required if the Cholesky factor R of X is given. From our computational experience, the Lanczos iteration can typically deliver a reasonably good estimate of $\lambda_1(B)$ (say, with an absolute error of less than 10^{-3}) in less than 20 Lanczos steps. Since each step of the Lanczos iteration takes about $4n^2$ flops for our matrix B, the total number of flops required in estimating $\lambda_1(B)$ is $\mathcal{O}(n^2)$, plus the cost of computing the Cholesky factor of X. Comparing the work required among the three methods discussed in this note, the proposed method is the cheapest.

We shall label the eigenvalues of B in decreasing order:

$$\lambda_1(B) \geq \lambda_2(B) \geq \lambda_3(B) \geq \cdots.$$

The vector 2-norm $\|\cdot\|$ will be used throughout.

2 Lanczos iteration

The Lanczos iteration [13, pp. 183–186] is an orthogonal projection method for estimating eigenvalues of a symmetric matrix. Suppose A is a general symmetric $n \times n$ matrix. Given an initial n-vector q_1 with $\|q_1\| = 1$, the Lanczos iteration constructs an $n \times j$ matrix $Q_j = [q_1\ q_2\ \cdots\ q_j]$ whose columns form an orthonormal basis for the Krylov subspace $\langle q_1, Aq_1, \ldots, A^{j-1}q_1\rangle$, successively for each j. In doing this, it also constructs a $j \times j$ symmetric tridiagonal matrix T_j that satisfies the equations:

$$AQ_j = Q_jT_j + t_{j+1,j}\, q_{j+1}e_j^T, \tag{2}$$

$$Q_j^TAQ_j = T_j, \tag{3}$$

where e_j denotes the jth standard unit vector in $\mathbb{R}^n$. It is known that the extremal eigenvalues of T_j are usually good approximations to the extremal eigenvalues of A even when j is substantially smaller than n. But the quality of approximations deteriorate as one proceeds from the extremal to the interior eigenvalues of T_j. This observation can be explained by the Kaniel-Paige convergence theory [9, p. 480] which established the rate at which each eigenvalue of T_j will converge to an eigenvalue

of A when j increases, together with a priori error bounds on the approximations. (Note that, however, these theoretical error bounds are not useful in practice since they depend on quantities that are not computable in the iteration process. For practical purpose, we need to use a posteriori error bounds that are presented in the next section.) The theory implied that the extremal eigenvalues of T_j will generally converge the fastest to the extremal eigenvalues of A. Thus for our matrix B, we can expect to estimate $\lambda_1(B)$ accurately by carrying out a small number of steps the Lanczos iteration.

To obtain the estimate of $\lambda_1(B)$, we compute the eigenvalues and eigenvectors of the $j \times j$ matrix T_j by the standard QR algorithm. The cost is only marginal when j is much smaller than n. Suppose the eigenvalues of T_j are labeled in decreasing order:

$$\tilde{\lambda}_1 \geq \tilde{\lambda}_2 \geq \tilde{\lambda}_3 \geq \cdots,$$

with corresponding eigenvectors (normalized to have unit norm)

$$\tilde{y}_1, \quad \tilde{y}_2, \quad \tilde{y}_3, \quad \cdots.$$

We use $\tilde{\lambda}_1$ to estimate $\lambda_1(B)$ and $\tilde{u}_1 := Q_j \tilde{y}_1$ as a corresponding approximate eigenvector for B. We do the same to estimate $\lambda_2(B)$ when necessary.

There are various ways to implement the Lanczos iteration. One of the most common implementation uses the 3-term recurrence relation to compute the orthonormal vectors, and the algorithm takes the following form [13]:

Choose a vector q_1 with $\|q_1\| = 1$.
For $j = 1, 2, \ldots,$
  $w = Aq_j$
  if $j > 1$
    $w = w - t_{j,j-1}\, q_{j-1}$
  end
  $t_{jj} = w^T q_j$
  $w = w - t_{jj}\, q_j$
  $t_{j+1,j} = \|w\| = t_{j,j+1}$
  $q_{j+1} = w/t_{j+1,j}$
end

Note that when A is the matrix $B = -R^{-T}\Delta X R^{-1}$, Aq_j can be computed by solving two triangular systems and a matrix-vector multiplication with a total cost of $4n^2$

flops. In fact, each step of the Lanczos iteration also costs about $4n^2$ flops, since all the other arithmetic operations cost at most $10n$ flops.

In exact arithmetic, the vectors $q_1, q_2, \ldots, q_{j+1}$ generated in the algorithm above are guaranteed to be mutually orthogonal. But in finite precision arithmetic however, they loss their global orthogonality very rapidly, and re-orthogonalization is required in order to maintain their orthogonality. For the ease of implementation, we use full re-orthogonalization when necessary. We refer the reader to [12] for partial re-orthogonalization techniques. The Lanczos iteration with full re-orthogonalization is as follows.

Choose a vector q_1 with $\|q_1\| = 1$.
For $j = 1, 2, \ldots,$
  $w = Aq_j$
  $\phi = \|w\|$
  if $j > 1$
    $w = w - t_{j,j-1}\, q_{j-1}$
  end
  $t_{jj} = w^T q_j$
  $w = w - t_{jj}\, q_j$
  if $\|w\| \leq 0.8\,\phi$
    For $i = 1, 2, \ldots, j,$
      $\eta = w^T q_i$
      $w = w - \eta\, q_i$
      $t_{ij} = t_{ij} + \eta$
    end
  end
  $t_{j+1,j} = \|w\| = t_{j,j+1}$
  $q_{j+1} = w/t_{j+1,j}$
end

For each j, the re-orthogonalization step incurs an extra cost of $4jn$ flops if it is performed. But if j is small, the extra cost incurred will not be significant compared to the cost of matrix-vector multiplication Bq_j.

3 Error bounds

Consider for $i = 1, 2$, the estimate $(\tilde{\lambda}_i, \tilde{u}_i)$ generated by the Lanczos iteration discussed in the last section for the eigen-pair corresponding to $\lambda_i(B)$ of the matrix B. We will

now establish a posteriori error bounds for the estimate of $\lambda_1(B)$. Let

$$r_i = B\tilde{u}_i - \tilde{\lambda}_i\tilde{u}_i, \qquad i = 1, 2. \tag{4}$$

(Note that $\|\tilde{u}_i\| = 1$.) Then the inequalities in the following proposition hold.

Proposition 3.1

$$\tilde{\lambda}_1 \leq \lambda_1(B) \leq \tilde{\lambda}_1 + \|r_1\|.$$

Proof. The inequalities follow from standard theorems in perturbation theory for eigenvalue analysis. The left-hand side inequality follows from the Courant-Fisher characterization [9, p. 411] of $\lambda_1(B)$ by noting that from (3), $\tilde{\lambda}_1$ is the Rayleigh quotient $\tilde{u}_1^T B\tilde{u}_1/\tilde{u}_1^T\tilde{u}_1$. The right-hand side inequality follows from the Bauer-Fike theorem [9, p. 342] for a symmetric matrix. □

A tighter a posteriori upper bound on $\lambda_1(B)$ is sometimes possible by appealing to the Kato-Temple theorem [11], [15]. The result is given next.

Proposition 3.2 *Suppose* $\tilde{\lambda}_1 > \tilde{\lambda}_2 + \|r_2\|$. *Then*

$$\lambda_1(B) \leq \tilde{\lambda}_1 + \frac{\|r_1\|^2}{\tilde{\lambda}_1 - \tilde{\lambda}_2 - \|r_2\|}.$$

Note the quadratic dependence on $\|r_1\|$.

Proof. Using the fact that $\lambda_2(B) \leq \tilde{\lambda}_2 + \|r_2\|$, the assumption of the proposition implies that $\tilde{\lambda}_1 \in (\lambda_2(B), \lambda_1(B)]$. Furthermore, $\tilde{\lambda}_1 - \lambda_2(B) > \tilde{\lambda}_1 - \tilde{\lambda}_2 - \|r_2\|$. By a corollary [13, p. 81] of the Kato-Temple theorem, the inequality in the proposition holds. □

Combining both propositions, we propose an upper bound ρ for $\lambda_1(B)$ as follows:

$$\rho = \begin{cases} \tilde{\lambda}_1 + \|r_1\|, & \text{if } \tilde{\lambda}_1 \leq \tilde{\lambda}_2 + \|r_2\|; \\ \tilde{\lambda}_1 + \min\left(\|r_1\|, \dfrac{\|r_1\|^2}{\tilde{\lambda}_1 - \tilde{\lambda}_2 - \|r_2\|}\right), & \text{otherwise.} \end{cases} \tag{5}$$

Thus if $\|r_1\|$ is a moderately small number (say, 10^{-3}), then ρ will be a good estimate of $\lambda_1(B)$ and hence $1/\rho$ will also provide a good estimate of the maximum step-length that can be taken from the interior-point iterate X.

Acknowledgments

Part of this research was done while the author was visiting the Department of Mathematical and Computing Sciences, Tokyo Institute of Technology, Tokyo, Japan. He thanks his host, Professor Masakazu Kojima, for providing an excellant research environment during his stint there. The author also thanks Professor Michael J. Todd for helpful discussions.

References

[1] F. Alizadeh, J. A. Haeberly, and M. Overton, *Primal-dual interior-point methods for semidefinite programming: convergence rates, stability and numerical results*, SIAM J. Optimization, 8 (1998), pp. 746–768.

[2] F. Alizadeh, J.-P A. Haeberly, M. V. Nayakkankuppam, M.L. Overton, and S. Schmieta, *SDPPACK user's guide*, Technical Report, Computer Science Department, NYU, New York, June 1997.

[3] S. J. Benson, Y. Ye, and X. Zhang, *Solving large-scale sparse semidefinite programs for combinatorial optimization*, working paper, Computational Optimization Laboratory, University of Iowa, May 1998.

[4] B. Borchers, *SDPLIB, A Library of Semidefinite Programming Test Problems*, available from `http://www.nmt.edu/~borchers/sdplib.html`.

[5] B. Borchers, *CSDP, a C library for semidefinite programming*, Technical report, New Mexico Tech., June 1998. Available at `http://www.nmt.edu/~borchers/csdp.html`.

[6] N. Brixius, F.A. Potra, and R. Sheng, *Solving semidefinite programming in Mathematica*, Reports on Computational Mathematics, No 97/1996, Department of Mathematics, University of Iowa, October, 1996. Available at `http://www.cs.uiowa.edu/~brixius/sdp.html`.

[7] N. Brixius, F. A. Potra, and R. Sheng, *SDPHA: A Matlab implementation of homogeneous interior-point algorithms for semidefinite programming*, available from `http://www.cs.uiowa.edu/~brixius/SDPHA/`, May 1998.

[8] K. Fujisawa, M. Kojima, and K. Nakata, *SDPA (semidefinite programming algorithm) user's manual — version 4.10*, Research Report, Department of Mathematical and Computing Science, Tokyo Institute of Technology,

Tokyo, May 1998. Available via anonymous ftp at `ftp.is.titech.ac.jp` in `pub/OpRes/software/SDPA`.

[9] G. H. Golub and C. F. Van Loan, *Matrix Computations*, 2nd ed., Johns Hopkins University Press, Baltimore, MD, 1989.

[10] C. Helmberg, F. Rendl, R. Vanderbei, and H. Wolkowicz, *An interior-point method for semidefinite programming*, SIAM J. Optimization, 6 (1996), pp. 342–361.

[11] T. Kato, *On the upper and lower bounds of eigenvalues*, J. Phys. Soc. Japan, 4 (1949), pp. 334–339.

[12] B.N. Parlett, *The Symmetric Eigenvalue Problem*, Prentice Hall, Englewood Cliffs, NJ, 1980.

[13] Y. Saad, *Numerical Methods for Large Eigenvalue Problems*, Manchester University Press, Manchester, UK, 1992.

[14] J. F. Sturm, *SeDuMi 1.00: Self-dual-minimization. Matlab 5 toolbox for optimization over symmetric cones*, Communications Research Laboratory, McMaster University, Hamilton, Canada, April 1998.

[15] G. Temple, The accuracy of Rayleigh's method of calculating the natural frequencies of vibrating systems, Proc. Roy. Soc. London Ser. A, 211 (1958), pp. 204–224.

[16] M. J. Todd, K. C. Toh, and R. H. Tütüncü, *On the Nesterov-Todd direction in semidefinite programming*, SIAM J. Optimization, 8 (1998), pp. 769–796.

[17] K. C. Toh, M. J. Todd, and R. H. Tütüncü, *SDPT3 — a* MATLAB *software package for semidefinite programming, version 1.3*, to appear in Optimization Methods and Software. A preliminary version of this paper had been announced through the Interior-Point Methods Online, and can be obtained through the worldwide web from `http://www.math.nus.sg/~mattohkc/index.html`.